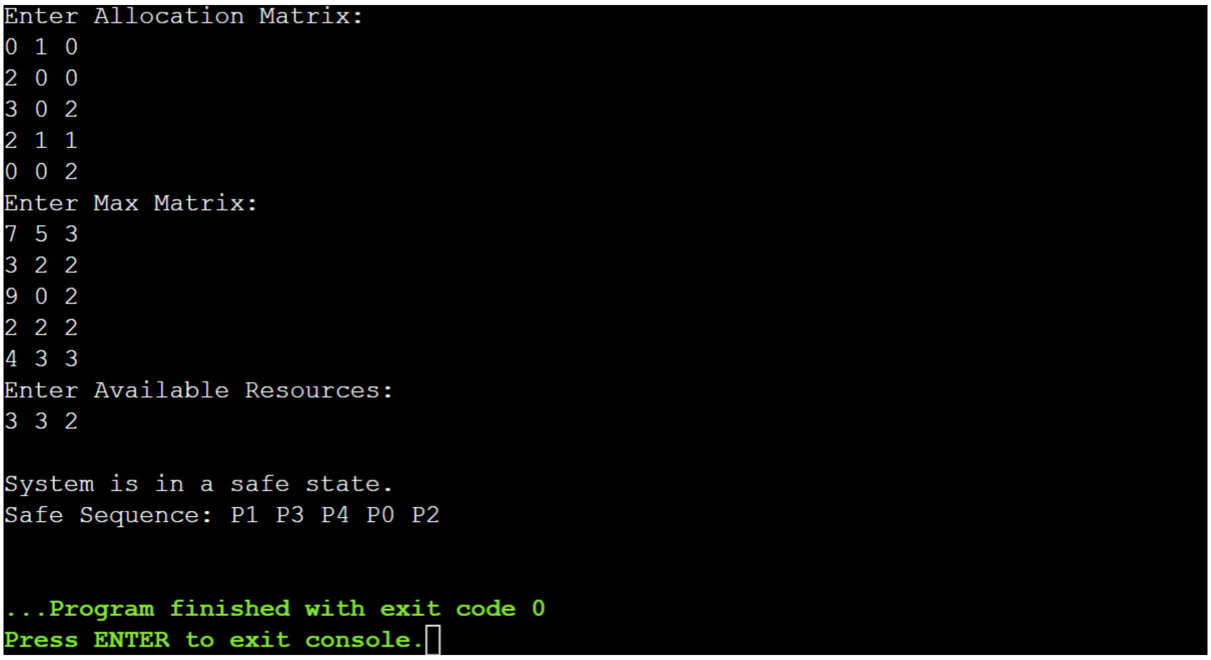
```
#include <stdio.h>

#include <stdbool.h>


#define MAX 10


int main() {

    int n, m, i, j, k;

    int Allocation[MAX][MAX], Max[MAX][MAX], Need[MAX][MAX];

    int Available[MAX];

    int Finish[MAX] = {0};

    int SafeSequence[MAX];


    printf("Enter number of processes: ");

    scanf("%d", &n);


    printf("Enter number of resource types: ");

    scanf("%d", &m);


    printf("Enter Allocation Matrix:\n");

    for (i = 0; i < n; i++)

        for (j = 0; j < m; j++)

            scanf("%d", &Allocation[i][j]);


    printf("Enter Max Matrix:\n");

    for (i = 0; i < n; i++)

        for (j = 0; j < m; j++)

            scanf("%d", &Max[i][j]);
```

```
printf("Enter Available Resources:\n");
for (j = 0; j < m; j++)
    scanf("%d", &Available[j]);

// Calculate Need Matrix
for (i = 0; i < n; i++)
    for (j = 0; j < m; j++)
        Need[i][j] = Max[i][j] - Allocation[i][j];

// Banker's Algorithm
int count = 0;
while (count < n) {
    bool found = false;
    for (i = 0; i < n; i++) {
        if (!Finish[i]) {
            for (j = 0; j < m; j++) {
                if (Need[i][j] > Available[j])
                    break;
            }
            if (j == m) {
                // Can allocate
                for (k = 0; k < m; k++)
                    Available[k] += Allocation[i][k];

                SafeSequence[count++] = i;
                Finish[i] = 1;
                found = true;
            }
            }
        }

        if (!found) {
            printf("\nSystem is not in a safe state.\n");
            return 1;
        }
    }

    printf("\nSystem is in a safe state.\nSafe Sequence: ");
    for (i = 0; i < n; i++)
        printf("P%d ", SafeSequence[i]);
    printf("\n");

    return 0;
}
```



```
Enter Allocation Matrix:
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2
Enter Max Matrix:
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
Enter Available Resources:
3 3 2

System is in a safe state.
Safe Sequence: P1 P3 P4 P0 P2


...Program finished with exit code 0
Press ENTER to exit console.
```